CG2111A Engineering Principle and Practice II

Semester 2 2022/2023

"Alex to the Rescue"

Design Report

Team: B02-4B

Name	Student #	Sub-Team	Role
Justin	A0257926N	Software	Motor Control
Zhong Heng	A0255320W	Hardware	Circuit design
Suveen	A0266706X	Software	Colour Sensor
Kiran	A0258805W	Hardware	Alex Assembly

Table of Contents

Section 1: System Functionalities

As part of the CG2111A Final Project, we will be designing an “Alex” robot which aims to perform a simulated Search and Rescue task, serving as a platform to locate “survivors” in a “disaster site” in place of people in precarious environments.

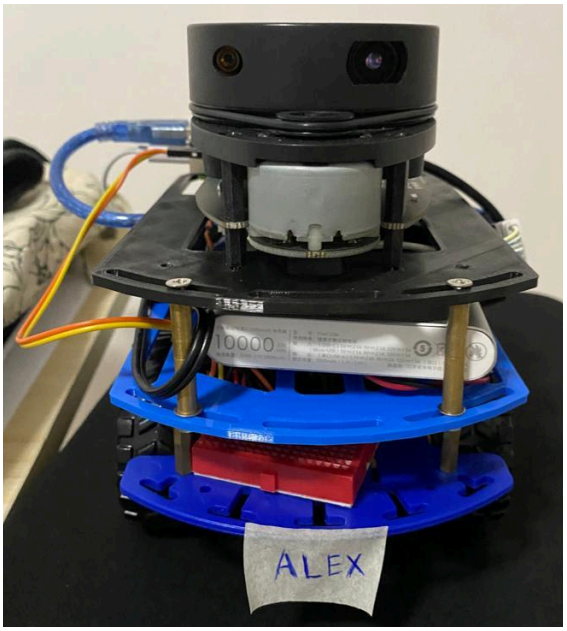

Figure I: Alex!

1.1 Problem Statement

Alex will be tele-operated using our laptop as a Ground Control Station (GCS) to navigate through a $3m^2$ area consisting of obstructions and walls (forming 2-4 rooms) of at least 18 cm in height. Alex’s goal would be to, within a time limit, navigate from the start room to the end room with the help of an operator, who will receive an environment map from Alex during the run. Alex should aim to perform this task as quickly as possible while minimising object collisions and maintaining fidelity of the generated environment map.

Additionally, specific objects demarking survivors can be found, of which the operator will have to identify and send a request to Alex to determine its colour using the provided colour sensor.

1.2 Hardware

Alex is designed with the Magician Robot Chassis and has a 2 wheel drive configuration. The motors are each connected to an encoder in order to determine the distance which each motor has moved. This helps us to achieve some form of odometry, albeit rather unreliable.

Power for Alex is split into power for the controllers and power for the motors. Controllers are powered using a power bank while the motors are powered via 4 rechargeable batteries.

Alex consists of an Arduino Uno microcontroller (hereforth referred to as Uno) and a RaspberryPi 4 single board computer (hereforth referred to as RasPi). The Uno is connected to a motor driver to control motor movement, along with the abovementioned encoders and a

forward pointed colour sensor. The RasPi is connected to the Uno via a USB B adaptor cable as well as a LIDAR module, the RPLIDAR-A1.

The schematic for the circuit centred around the Uno (since the RasPi is only connected to other devices via USB) is provided below.

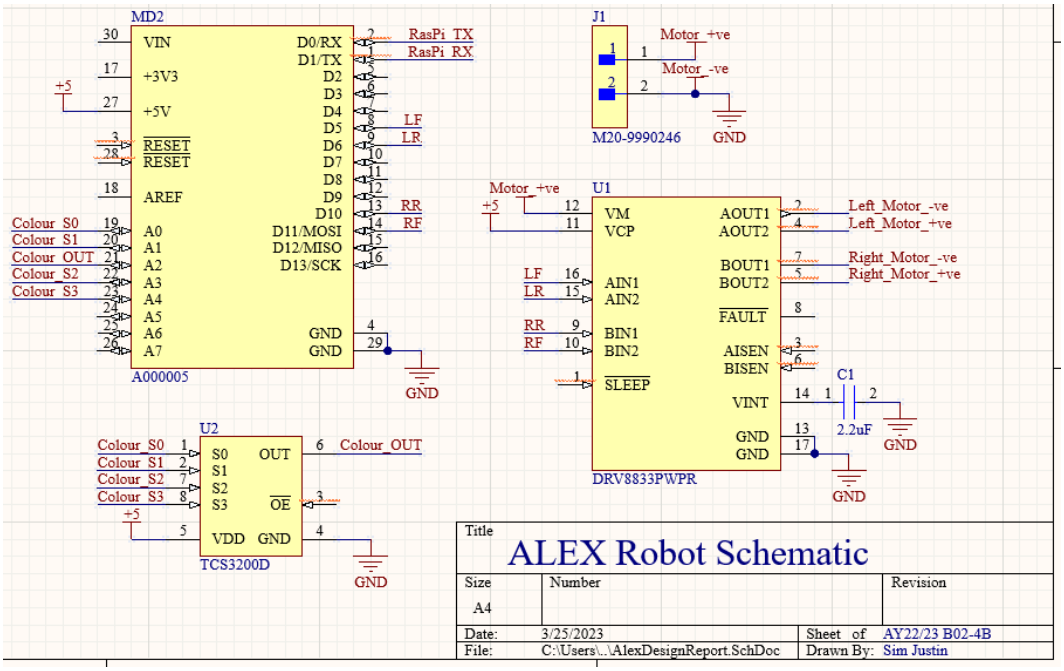

Figure II: Schematic Diagram for Alex

1.3 Software

The GCS will be the means through which the operator will handle Alex. Commands will be sent from the GCS to Alex remotely via the Internet, making use of the concept of secure communications (to prevent hacking of Alex!).

The RasPi runs the master control program (MCP) for Alex. Data going to and coming from the GCS and the Uno will be compiled in this program, along with data from the LIDAR. The RasPi will run on ROS Noetic for Debian 10 Buster which can be used to subscribe to messages from the LIDAR module and its SLAM algorithm, of which the one we are looking to implement will be the Hector SLAM algorithm. Communication with the Uno will be done serially via the USB B cable.

The Uno is a peripheral module used for controlling the motors and receiving data from the colour sensor. It interfaces with the RasPi, providing it with odometric and colour data from the aforementioned sensors.

A UML diagram for the data flow can be found in Section 3.

1.4 Additional Functionalities

2 areas for personal improvement can be performed to enhance the capabilities of Alex as suggested in the specifications manual.

1.4.1 "Miss Scarlet" is Alive!

Colour objects placed within the maze simulate targets for identification which we will be able to see on our map to send commands to Alex to use the colour sensor to recognise the colour of the object, after which Alex will relay the information back to the GCS. The abovementioned colour sensor will be used for this task.

1.4.2 Hardware Improvements: Limit Switch

We have decided to implement a limit switch at the bottom of the front hull which can be used for early detection of the hump. This will enable us to locate the hump and avoid it to prevent Alex from getting stuck on low lying obstacles.

1.4.3 Software Improvements: Return to Base

A search and rescue robot has to be able to return to its operator after completing its mission. To tackle this, we intend to implement an autonomous return to base function making use of the PNG map generated by the SLAM algorithm to generate a patch to return to base from Alex's current position.

1.4.4 Software Improvements: Modelling Alex in ROS

The implementation of Hector SLAM which we are provided from the labs only allow us to attain the odometry of the LIDAR module and does not enable us to visualise the boundaries of the robot. Naturally, Alex is larger than the footprint of the LIDAR module. To make sure that we do not end up bumping into walls and debris, we will be attempting to come up with a rudimentary model of Alex in ROS such that we will be able to visualise Alex in its environment through rviz.

Section 2: Review of State of the Art

2.1 Search and Rescue Robot with Tele-Operated Tether Docking System (RAPOSA)

We explore RAPOSA developed by IDMind and Institute for Systems and Robotics (Marques et al., 2007, 332) in our 1st case study.

2.1.1 Functionalities

RAPOSA is a tethered tracked robot designed for Urban Search and Rescue designed to operate in outdoor environments unsafe for human traversal, aiming to locate potential survivors of which information will be fed back to the remote human operator.

2.1.2 Hardware and Software

RAPOSA's has a 2 segment chassis with track-based locomotion, allowing for traversal of rough terrain (Zhu et al., 2018, 1). Peripherals mounted on RAPOSA include a thermal camera, 2 vector-controlled optical cameras and gas, temperature, humidity sensors and LEDs, a microphone and a loudspeaker. Actuation comprises a robotic arm controlled by multiple motors and sensors. To facilitate tethered operation, a cable is attached.

RAPOSA is tele-operated using a remote control unit, allowing for real-time control of its traversal and actuation. AI algorithms aid its navigation and object recognition, enabling it to make decisions based on sensor inputs and environmental conditions. The control unit includes a software providing real-time monitoring and data management to the operator.

2.1.3 Strengths and Weaknesses

RAPOSA's use of a tether eliminates its need for battery charging while its sensors and actuators enable it to perform necessary tasks during search and rescue emergencies.

RAPOSA's tether however also limits its mobility and range and the requirement for on-the-site operators may be problematic if trained operators are unavailable.

2.2 Airobotics Optimus

Our 2nd case study looks at Airbotics Optimus developed by ONDAS (ONDAS, n.d.).

2.1.1 Functionalities

Airobotics Optimus is a tele-operated UAV built for search and rescue operations targeted at disaster assessment and preliminary scene assessment.

2.1.2 Hardware and Software

Propulsion is achieved using a quadrotor setup and it is built with a variety of sensors including a gimbaled high-res camera and LIDAR. It is capable of wireless communication with a GCS.

AI improves its capabilities (i.e. autonomous flight, take-off, landing, collision avoidance, mapping). It uses a proprietary OS allowing for real-time flight control, autonomous operations, and remote monitoring along with storage and processing of sensor data in real-time or in the future.

2.1.3 Strengths and Weaknesses

Airobotics Optimus features accurate sensors for precise mapping and identification of targets. Its AI capabilities allow for some degree of autonomous movement as well.

However, it is a costly system to operate and its flight time may limit its range.

Section 3: System Architecture

3.1 Alex UML Diagram

A UML deployment diagram for Alex is provided below. This highlights the connections between the various devices, notably the 3 key devices being the GCS (laptop), the RasPi and the Uno, as well as the flow of data between the various modules.

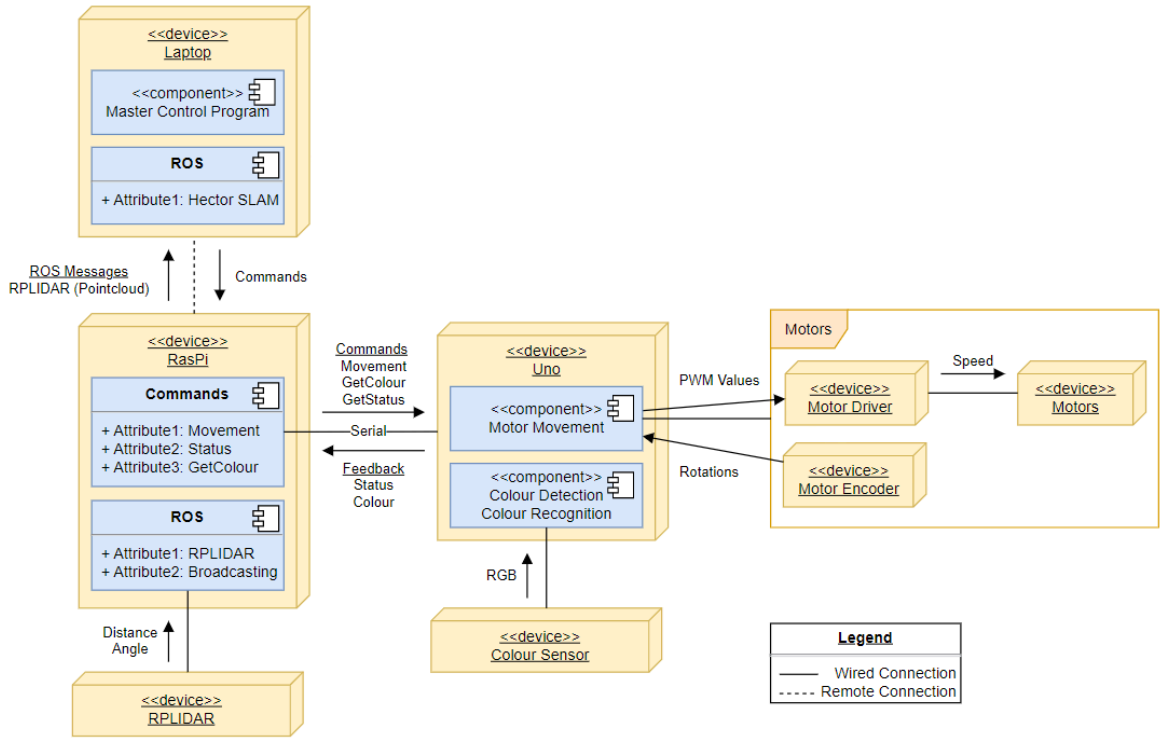

Figure III: UML Diagram for Alex System

Further information with regards to algorithm design and flow can be found in the subsequent section on Component Design.

3.2 Hardware Design Considerations

To aid our quest in accomplishing Alex's task, we have made a few hardware design considerations when building Alex. Given that Alex has a two-wheel drive configuration, we placed the LIDAR module on the axis of rotation such that there will be minimal translation of the module during turning, allowing for greater ease of obtaining a map of the desired area.

Additionally, we have 3D printed two layers in Alex (light blue middle layer and black top layer) to aid us in our design. 3D printing provides us with more agency in placement of components aboard Alex, providing greater ease of wire and device management.

3.3 Software Design Choices

We decided to make certain key software design choices in order to resolve potential problems. One of which is the decision to offload computation from the RasPi onto the GCS, such as in the case of the implementation of the SLAM algorithm. While this may affect real time capability of Alex, we believe that at the low speeds at which Alex is travelling, this would not pose a significant issue to our use case. Offloading computation onto the GCS is done in order to minimise CPU load on the RasPi to prevent thermal issues from occurring.

Section 4: Component Design

In this section we highlight the algorithm governing Alex in its completion of the main task as set out in Section 1, excluding part 4 on additional functionalities as design for those modules are works in progress.

4.1 Algorithm Overview

1. Initialisation
2. Receive User Command
3. Perform Command
4. Repeat Step 2 until Navigation is Complete

4.1.1 Initialisation

1. Start remote communication between GCS and RasPi
2. Check serial status for communication between RasPi and Uno
3. Start ROS and necessary launch files for RPLIDAR setup
4. Begin ROS broadcasting from RasPi to GCS (pointcloud data from LIDAR)
5. Spin around to acquire initial surrounding conditions

4.1.2 Receive User Command

1. Packet sent to RasPi
    a. Decodes command type
    b. Forward packet to Uno if necessary
2. RasPi sends acknowledgement to GCS
3. Uno sends acknowledgement to RasPi

4.1.3 Perform Command

1. Types of Commands: Movement, Status, GetObject, End
    a. Movement: Actuate motors until stop condition (target distance reached or stop command issued)
    b. Status: RasPi retrieves data about movement
    c. GetObject: Arduino reads from colour sensor, computes the colour of the object and sends the colour back to the RasPi (in turn the GCS)

4.1.4 Repeat until Navigation is Complete

Repetition of step 2 and 3 until navigation is complete

Upon receiving “End” command, Alex will stop all activities

Section 5: Project Plan

We present the timeline of our internal deliverables and milestones below.

Week	Milestones	Hardware Deliverables	Software Deliverables
Week 8			Remote Control
Week 9			Secure Communications
Week 10		Complete Base Assembly	Test ROS & SLAM
Week 11	Design Report Due	Implementation of Limit Switch	Implement Colour Detection Integration of Movement, Communications and LIDAR Modules Minimise RasPi Load
Week 12	Trial Run		Return to Base Modelling Alex in ROS
Week 13	Final Demo		Final Testing
Reading Week	Final Report Due		

Table I: Timeline for Internal Deliverables

Deadlines highlighted above are a result of careful time management with extra consideration being put towards balance of time with examinations preparation and deadlines from other modules which our members are (begrudgingly or not) reading.

References

Marques, C., Cristóvão, J., Alvito, P., Lima, P. U., Frazão, J., Ribeiro, I., & Ventura, R. (2007, June). A search and rescue robot with tele-operated tether docking system. *Industrial Robot: An International Journal*, *34*(4), 332 - 338. ResearchGate. 10.1108/01439910710749663

ONDAS. (n.d.). *Optimus*. Airobotics. Retrieved March 23, 2023, from https://www.airoboticsdrones.com/optimus/

Zhu, Y., Kan, J., Li, W., & Kang, F. (2018, May 13). Strategies of traversing obstacles and the simulation for a forestry chassis. *International Journal of Advanced Robotic Systems*, *15*(3), 1 - 13. SAGE journals. 10.1177/1729881418773903